1. Giới thiệu Express:
   1. Express là gì ?
      * Express js là một Framework nhỏ, nhưng linh hoạt được xây dựng trên nền tảng của Nodejs. Nó cung cấp các tính năng mạnh mẽ để phát triển web hoặc mobile
      * Về các package hỗ trợ: Expressjs có vô số các package hỗ trợ nên các bạn không phải lo lắng khi làm việc với Framework này.
      * Về performance: Express cung cấp thêm về các tính năng (feature) để dev lập trình tốt hơn. Chứ không làm giảm tốc độ của NodeJS.
      * Và hơn hết, các Framework nổi tiếng của NodeJS hiện nay đều sử dụng ExpressJS như một core function, chẳng hạn: SailsJS, MEAN,…
   2. Hướng dẫn cài đặt:
      * Do Express framework không được tích hợp sẵn vào trong node.js nên khi muốn sử dụng nó chúng ta phải tải nó về máy với sử dụng được. Và để cải đặt nó chúng ta sử dụng cú pháp của npm: npm install express –save
   3. Các Route trong Express:
      * Route GET method: Router này chỉ nhận phương thức get đến url.

**VD**: trả về chữ 'hello world' khi người dùng get request vào url có param là say.

app.get(‘/say’, function(req,res){

res.send(‘hello word’);

});

* + - Route POST method: Router này chỉ nhận phương thức post.

**VD**: Trả về chữ 'hello world' khi người dùng gửi post request đến url có param là say.

app.post(‘/say’, function(req,res){

res.send(‘hello word’);

});

* + - Các Method khác: Đối với các method khác cung làm tương tự như hai method get và post. Chỉ việc đổi tên method thành method mà bạn muốn nhận.

//put method

app.put(‘/say’, function(req,res){

res.send(‘hello word’);

});

//delete method

app.delete (‘/say’, function(req,res){

res.send(‘hello word’);

});

* + - Nhận tất cả các Method: Trong trường hợp các bạn muốn nhận tất cả các method thì chỉ cần đổi method thành all là ứng dụng sẽ nhận tất cả các method.

app.all(‘/say’, function(req,res){

res.send(‘hello word’);

});

* + - Truyền tham số vào router: Để truyền tham số vào router chúng ta sẽ sử dụng dấu : và tên biến vào path.

app.get(‘/user/:name, function(req,res){

res.send(‘User có tên là: ‘ + req.params.name);

});

* 1. Views and Layouts: Trong Express hỗ trợ nhiều Template engine dùng cho việc tạo Views nhanh chóng và cần thiết cho app

var app = express();

// set up template engine

app.set(‘view engine’, ‘ejs’);

app.set(‘views’, ‘./Views/page’);

* 1. Static Files and Views: Các tệp và dạng xem tĩnh Express dựa trên một phần mềm trung gian để xử lý các tệp và chế độ xem tĩnh. Phần mềm trung gian tĩnh cho phép bạn chỉ định một hoặc nhiều thư mục chứa tài nguyên tĩnh đơn giản được gửi tới máy khách mà không cần xử lý đặc biệt. Đây là nơi bạn sẽ đặt những thứ như hình ảnh, tệp CSS và tệp JavaScript phía máy khách. Trong thư mục dự án của bạn, tạo một thư mục con được gọi là public (chúng ta gọi nó là public bởi vì bất cứ thứ gì trong thư mục này sẽ được phục vụ cho client). Sau đó, trước khi bạn khai báo bất kỳ tuyến nào, bạn sẽ thêm phần mềm trung gian tĩnh:

Khai báo:

app.use(express.static(\_dirname + ‘/public’));

vd:

<img src=”/img/logo.png”>

* 1. Ngoài ra: nó còn giúp chúng ta truyền được những dữ liệu vào Views một cách dễ dàng:

Phần view: about.ejs

<%= name %>

Phần controler:

app.get(‘/’, function(req,res){

res.render(‘about’, {name: “Hello”});

});

## Thành phần của một đường dẫn URL

Đây là các thành phần chính của URL này :

* Giao thức là http. Ngoài ra còn có nhiều giao thức khác như https, ftp, v.v;
* Tên host hay hostname là video.google.co.uk;
* Tên miền thứ cấp là video;
* Tên miền chính là google.co.uk;
* Tên miên cấp 1 (top-level domain) la uk.Tên miền uk là tên miền quốc gia
* Tên miền cấp 2 (second-level domain) là co.uk;
* Cổng giao tiếp là 80, đây là cổng ngầm định của dịch vụ Web. Có thể sử dụng các cổng khác như 8080 hay 8000. Khi cổng là 80, phần lớn người ta không cần thêm vào đường dẫn;
* Đường dẫn là /videoplay, giúp để trỏ đến một tệp tin hay một vị trí nằm trên máy chủ Web;
* Đường dẫn URL nói trên ẩn chứa các tham số. Tên của tham số là docid và giá trị của nó là -7246927612831078230. Đường dẫn có thể có nhiều tham số. Tham số đầu tiên bắt đầu bằng dấu hỏi (?), các tham số sau nối tiếp bằng dấu và (&);
* Ký tự “#00h02m30s” còn được gọi là neo dùng để trỏ đến một vị trí trên văn bản hiện tại. Trong trường hợp trên vị trí neo trỏ đến đoạn phim thời điểm 2 phút, 30 giây trong đoạn video tương ứng.

1. The Request and Response Objects:

Nếu như lập trình front-end quan tâm đến giao diện, đến trải nghiệm người dùng, đến HTML, CSS và JavaScript, thì lập trình back-end quan tâm đến việc chọn lọc và gửi đi dữ liệu chứa trong database phù hợp với yêu cầu của người dùng.

Do phía server liên tục:

* nhận các HTTP request từ trình duyệt,
* xử lý các HTTP request
* lôi dữ liệu từ database ra xử lý theo yêu cầu
* tạo HTTP response để phản hồi cho trình duyệt

cho nên có 3 đối tượng chính mà lập trình back-end cần học để làm việc:

1. HTTP request
2. HTTP response
3. Data trong database

HTTP resquest :

* HTTP là viết tắt của 1 giao thức, tạm gọi đây là quy định về giao tiếp giữa server với trình duyệt.
* request là yêu cầu. Tức là 1 yêu cầu của trình duyệt đòi server gửi thông tin để render 1 trang web, hoặc gửi 1 file nhạc, file ảnh.

Đã là giao thức, thì bao giờ cũng phải có các tiêu chuẩn quy định , có thể thì mới giúp 2 bên hiểu nhau trong nhiều tình huống, không bị hiểu nhầm, hiểu sai.

Trong mỗi HTTP request có gì? Có 1 vài thứ:

* có URL để trỏ đến tài nguyên trên web server mà trình duyệt đang cần. Tài nguyên là gì? file HTML, JS, CSS, hoặc thậm chí 1 file JSON.
* bản chất của HTTP request có vài loại, như GET, POST, PUT, UDPATE, DELETE, HEAD. Mỗi cái tương ứng với 1 loại công việc mà trình duyệt yêu cầu web server đáp ứng.
* data được encoded bên trong body của HTTP request.

HTTP request thực chất là 1 file text. Tùy vào từng loại HTTP request mà file text khác nhau ít nhiều. Tuy vậy, trong thực tế, người ta dùng dạng POST trong trường hợp người dùng submit form để gửi thông tin đi, do dạng POST encode thông tin vào body của request thay vì để nó ra ngoài URL giống như GET

### HTTP response

HTTP response thì mang trong mình nó 2 thứ:

* Code status, để báo hiệu cho trình duyệt biết là request nó gửi đi được xử lý thành công hay không.
* Dữ liệu mà trình duyệt cần.

Template Engine

Có rất nhiều template engine được sử dụng cho Nodejs. Nếu chúng ta sử dụng Express.js thì có thể cái plug-in consolidate.js nó hỗ trợ rất nhiều template engine, có thể kể ra danh sách template engine như sau.

* atpl
* dust (website)
* eco
* ect (website)
* ejs
* haml (website)
* haml-coffee (website)
* handlebars (website)
* hogan (website)
* jade (website)
* jazz
* jqtpl (website)
* JUST
* liquor
* mustache
* QEJS
* swig (website)
* templayed
* toffee
* underscore (website)
* walrus (website)
* whiskers

Nhưng với việc có rất nhiều template engine thì sẽ khiến chúng ta phân vân không biết nên chọn cái nào. Để trả lời cho câu hỏi cái nào sẽ là template engine tốt nhất cho Nodejs? Câu trả lời có lẽ là không có, vì nếu đã có một template engine tốt nhất rồi thì tại sao lại còn có những template engine khác làm gì. Ở đây chúng ta chỉ có thể trả lời cho việc Template engine nào được sử dụng nhiều nhất, và điểm mạnh, điểm yếu của mỗi cái để người dùng có thể chọn lựa sao cho phù hợp với dự án.

# Những Template Engine phổ biến

Theo như nghiên cứu tài liệu từ một số chỗ, chẳng hạn như Jade có vẻ như là template engine được dùng phổ biến nhất, nó cũng là template engine mặc định khi mà cài đặt Express.js, nhưng có vẻ như khá nhiều khá thích sử dụng EJS bởi vì tính thuận tiện và đơn giản

# EJS

Nếu như bạn đã quen thuộc với việc sử dụng HTML thì việc dùng EJS khá dễ dàng, vì với syntax khá giống với html, bạn có thể tạo ra các partial view với ejs để giúp cho việc reused trở nên dễ dàng và giúp cho code dễ đọc, dễ bảo trì hơn. EJS giúp bạn truyền data vào các views. Một số bước để có sử dụng EJS với Nodejs

## Cài đặt

chỉ cần chạy lệnh

$ npm install ejs

## Sử dụng

var express = require('express');

var app = express();

app.set('view engine', 'ejs');

app.get('/', function(req, res){

res.render('index',{user: "Great User",title:"homepage"});

});

Chúng ta cần bước set view Engine cho ejs

app.set('view engine', 'ejs');

vì mặc định khi cái express.js thì default view engine sẽ là Jade(pug) Chúng ta truyền view tới user bằng cú pháp

res.render('index',{user: "Great User",title:"homepage"});

với index là view và dữ liệu được truyền là user và title. Với việc tạo và sử dụng Partial views giả sử chúng ta đã tạo ra 2 file là header.ejs

<title>

Hello World

</title>

và body.ejs

<div>

Welcome, Great User

</div>

thì ở chỗ gọi tới 2 file đó chúng ta dùng từ khóa include

<html>

<head>

<% include('header.ejs') %>

</head>

<body>

<% include('body.ejs') %>

</body>

</html>

Chúng ta có thể sử dụng javascript cho tempalating trong ejs Ở trong ejs không có hỗ trợ block, mà những việc thực hiện với các danh sách chúng ta phải sử dụng forEach

<ul>

<% users.forEach(function(user){ %>

<%= user.name %>

<% })%>

</ul>

Với data tương ứng cho danh sách như sau

app.get('/', function(req, res){

res.render('index',{users : [

{ name: 'name1' },

{ name: 'name2' },

{ name: 'name3' }

]});

});